Improved Merkle Cryptosystem (IMC)

Attila Altay Yavuz[1], Emin Anarim[2], and Fatih Alagoz[1]

[1] Bogazici University, Department of Computer Engineering,
Bebek, Istanbul 34342, Turkey
[2] Bogazici University, Department of Electrical and Electronic Engineering,
Bebek, Istanbul 80815, Turkey
{attila.yavuz, fatih.alagoz}@boun.edu.tr
anarim@boun.edu.tr

Abstract. Merkle Cryptosystem (MC) is the first cryptosystem which introduces general concept of the public key cryptography. In this paper, we propose Improved Merkle Cryptosystem (IMC), which has significant security advantages over both MC and a variant of MC (VMC). In IMC, cryptographic hash functions and a new puzzle structure are used together in order to increase the security of MC and VMC. The key agreement value, which is send as clear text in VMC, is hidden using cryptographic hash function in IMC. Also, in order to increase security of the key agreement value, auxiliary keys are used. Notice that, in IMC, computational advantages of VMC remain unchanged while its security is increased. Utilizing computational advantages of VMC, IMC has also security and storage advantages over original MC. It is shown that, with these improvements, IMC can provide as high security as some well-known public key cryptosystems while MC and VMC can not provide same security due to performance problems.

Keywords: Cryptography, Merkle Cryptosystem, Key Establishment, Encryption.

1 Introduction

Public key cryptography made significant impact on secure and authenticated communication systems [1]. Many different public key cryptography algorithms have been developed based on different mathematical approaches [2]. RSA, which is based on factorization of large numbers into prime factors and El-Gamal cryptosystem, which is based on hardness of Discrete Logarithm Problem (DLP), are well-known and fundamental public key cryptosystems [3]. Also, Elliptic Curve Cryptography (ECC) [4] which is based on DLP over EC is one the most widely used cryptosystem utilizing DLP. Apart from these, new public key cryptosystems such as NTRU (N-th degree TRUncated polynomial ring) [5], which is based on lattice problem, have also been proposed.

However, the first cryptosystem, which provides a solution to the secure communication problem over insecure channels without pre-established secrets, is Merkle Cryptosystem (MC) [6]. In MC, communicating parties use 'puzzles',

A. Levi et al. (Eds.): ISCIS 2006, LNCS 4263, pp. 924–934, 2006.

which are feasible for them to solve but infeasible for an attacker to solve. A Variant of MC (VMC) [7] utilizes MC with block ciphers and uses a different puzzle generation technique from MC. VMC method has some advantages over original MC.

In this paper, we propose Improved Merkle Cryptosystem (IMC) that increases the security of the both MC and VMC. In VMC, the index, which is used for key agreement, is sent in clear text. This approach causes significant security degradation. In IMC, we use a different puzzle structure and cryptographic hash functions to increase security of VMC. IMC utilizes puzzle generation method of VMC but uses auxiliary key to increase security of messages transmitted over network. Also, in order to hide key agreement value, we use cryptographic hash functions. Thus, adversary can not understand which puzzle communicating parties agree on (auxiliary keys increases security of the hashed key agreement value). In addition to this, computational advantages of the VMC remain unchanged while its security is significantly increased. Shifting computational advantages of communicating parties to the overall system security, IMC can also provide higher security than original MC. Moreover, puzzle structure of IMC provides storage advantages over original MC method. We also show that, with these improvements, IMC can provide as high security as some well-known public key cryptosystems while MC and VMC can not provide same security due to performance issues.

The rest of the paper is organized as follows: In Section 2, we discussed MC and VMC algorithms together with their security analysis. In Section 3, we present our IMC algorithm and its properties. In Section 4, we give detailed analysis of IMC algorithm and compare IMC to MC and VMC. Also, comparison of IMC for various criteria to some well-known public key cryptosystems and MC-VMC is given. In Section 5, we present conclusion and future works.

2 MC and VMC

2.1 Merkle Cryptosystem (MC)

Merkle Cryptosystem (MC), also known as Merkle Puzzle, is the first cryptosystem having public key cryptography properties [6]. Suppose that, Alice and Bob want to secretly communicate over an insecure channel without pre-established secrets. Alice creates a set of puzzles that are feasible to solve for Bob. These puzzles are derived from secre*t* values using secret keys that are short enough such that Bob can realize brute force attack on them. Each puzzle contains a session key that will be used for future communication and a pseudo-index which makes possible secret key establishment. In addition, each puzzle is added the required redundancy that allows Bob to perform the brute force attack. Bob selects one of the puzzles and performs a brute force attack on it. Bob stops brute force attack when he detects recognizable redundancy value. Bob recovers pseudo-index and session key from solved puzzle and sends pseudo-index to Alice. Alice searches this pseudo-index in her pseudo-index list and find corresponding real index which Bob has chosen. Consequently, Bob and Alice agree on a secret

session key which corresponds to the selected real index. Adversary (Oscar) only obverses pseudo-index, which does not reveal any information about which key Bob has chosen. Thus, adversary has to make brute force attack to all puzzles. Here, Bob makes brute force attack only one puzzle while adversary makes brute force attack to all puzzles.

Apart from being the first cryptosystem which introduces general concept of the public key cryptography, principles of MC are used in many security applications. For instance, puzzle principle of MC is used in time-lock puzzles [8]. In time-lock puzzles, the idea is that a secret is transformed in such a way that any machines, running continuously, take at least a certain amount of time to solve the puzzles in order to recover the secret. This minimum amount of time is the relative release time with respect to the start of solving the puzzle and could be different for different machines [9]. In addition to this, puzzle concept of MC is used to combat against junk mails and is used to prevent DoS (Denial of Service) attacks utilizing client puzzles [10].

Notations, which are used in MC, are given below:

P : Public key vector (puzzles), K : Secret key vector that is used to generate P, K_s : Session key vector. $P_i \in P$, $K_i \in K$ and $K_{s_i} \in K_s$ for $1 \leq i \leq N$ where $N = 2^m$. m: The parameter which, determines number of elements in the P, K and K_s vectors. $|Var|$ denotes the bit length of the variable Var and $||$ denotes concatenation operation. In MC, bit length of the secret key is represented by $n' = |K_i|$ and bit length of the single puzzle is represented by $t = |P_i|$. $(E-D)_K$: Symmetric encryption and decryption functions using secret key K. r : Index number used for key agreement. S : The recognizable redundancy value.

Original version of MC is described below:

1. Alice generates puzzles $P_i = E_{K_i}(S||r_i||K_{s_i})$ for $1 \leq i \leq N$ where $N = 2^m$. Alice sends vector P to Bob.

2. Bob selects one of the puzzle say j'th puzzle and realizes a brute force attack to P_j. When brute force attack is completed, Bob decrypts the puzzle $(S||r_j||K_{s_j}) = D_{K_j}(P_j)$. Bob verifies S and recover r_j and K_{s_j}.

3. Bob sends index r_j to Alice in clear text. Notice that this index r_j is a pseudo-index and only Alice knows which real index corresponds to pseudo-index r_j. Suppose that pseudo-index r_j corresponds $i'th$ puzzle. Then, Alice knows that Bob has chosen $i'th$ puzzle P_i from P.

4. Alice and Bob agree on the secret session key K_{s_i} and use this key for future communication.

In this system, symmetric encryption function can be an appropriate block cipher such as DES or AES [11]. Notice that, n' i.e., that is the bit length of the K_i should be selected carefully. It should allow Bob to realize a brute force attack on P_i but should not be so small such that it weakens the whole cryptosystem. In the first version of the MC, n' is selected as 20 bits. Also, some versions select $n' = m$ so that number of puzzles and bit length of the single puzzle are equal to each other. In MC, Oscar can listen the communication channel and observe index r_j . However, since index r_j does not reveal which puzzle Bob chooses, Oscar has to realize brute force attack to whole puzzles in order to understand

which puzzle Bob has chosen. In original MC, bit length of single puzzle P_i is $|t| = (S||r_i||K_{s_i})$ where $t > n'$. This property increases storage requirements of the original MC. Notice that, reasonable bit length of $n' \leq 50$. Complexity of the MC is summarized at Table 1.

Table 1. Computational and Storage Complexity of MC

	Computational Complexity	Storage Complexity
Alice	$O(\ 2^m\ )$	$O(2^m) * t$
Bob	$O(\ 2^n\ )$	
Oscar	$O(\ 2^{m+n}\ )$	

2.2 Variant of Merkle Cryptosystem (VMC)

Many variations of MC have been reported in literature. One of the variant (VMC), which is given in [7], uses larger key bit length for each puzzle. Also, puzzle generation method of VMC is different from MC. For this reason, it is not feasible for Bob to attack each puzzle similar to the original MC. This method uses another public key X to generate public key vector such that length of the public key vector can be used to reduce search space of the participant of the communication. However, this method sends real index in clear text that significantly reduces brute force attack effort of Oscar. Parameters, which are used in VMC, are given below:

X : The public key value, which is used to generate public key vector P. Bit length of the single puzzle is equal to bit length of the secret key, $|P_i| = |K_i| = n$. Notice that, reasonable bit length of the $n \approx 64$ bits.

VMC algorithm is described below:

1. Alice generates puzzles $P_i = E_{K_i}(X)$ for $1 \leq i \leq N$ where $N = 2^m$. Alice sends vector P and public key X to Bob.

2. Bob generates random keys $l_1, l_2, ...$ and encrypts X with them. Bob compares results with elements in vector P such that there is a collision between encrypted value and one of the elements of vector P. Suppose that collision occurs for l_j. Consequently, $E_{l_j}(X) = P_{K_i}(X)$ and $l_j = K_i$. Bob finds the i'th puzzle in vector P via this collision search.

3. Bob sends index i in clear text together with encrypted message $M' = E_{K_i}(M)$. Bob sends (M', i) to Alice. Optionally, Bob might generate a session key K_s, $K_s' = E_{K_i}(K_s)$ and sends (K_s', i) to Alice.

4. Alice obtains index i and understands that Bob uses i'th key for secret communication. Alice decrypts message or session key $M = D_{K_i}(M')$ or $K_s = D_{K_i}(K_s')$.

Complexity of the VMC is summarized at Table 2.

Note that, VMC uses block ciphers to generate puzzles. The bit length of a single puzzle (n) is smaller than the key bit length of the block cipher such as AES having 128,192 or 256 bit key size in order to make collision search

Table 2. Computational and Storage Complexity of VMC

	Computational Complexity	Storage Complexity
Alice	$O(2^m)$	$O(2^m) * n$
Bob	$O(2^{n-m})$	
Oscar	$O(2^n)$	

possible for Bob. In this situation, first n bit of the block cipher key is used as variable part while remainder part is used as constant to obtain n bit security. In remainder of the paper, n is used in this context.

2.3 Advantages and Disadvantages of VMC to MC

Most important contribution of VMC is that the computational effort of Bob to find a secret key, is reduced from $O(\ 2^{n'}\ )$ to $O(2^{n-m})$. The reason is that, Bob realizes a collision search using advantages of large number of puzzles. In collision search, as explained in VMC step 2, Bob generates random keys and discovers corresponding private key with the probability of $Pr(Collision) = 2^m/2^n$. However, in original MC, number of puzzles does not give any contribution for reducing computational effort of Bob. The reason is that, Bob directly chooses one of the puzzles and realizes a brute force attack to the puzzle with $O(2^n)$ computational complexity.

VMC uses computational advantages of Bob to generate puzzles that have larger key bit length. Instead of giving Bob shorter time to determine a key, it is possible to keep collision search to constant but increase the bit length of single puzzle n. This approach increases computational effort of Oscar to break a single puzzle.

In VMC, Bob sends key agreement index i in clear text together with encrypted message as described in VMC step 3. Sending index in clear text causes significant security problem and drastically reduces the computational effort that Oscar has to perform, compared to the original MC. Notice that, in MC, Bob sends his selected index in clear text but the index sent by Bob does not correspond to the real index for the puzzle that Bob has found. Alice generates a puzzle in the MC step 1 such that it contains a pseudo-index which is known only by Alice. In MC Step 3, this pseudo-index is sent in clear text and does not reveal any information about the real index that Bob has found. Thus, brute force attack effort of Oscar is in the order of $O(2^{n+m})$. However, in VMC, sending real index i in clear text (Step 3) reduces the effort required to attack by Oscar from $O(2^{n+m})$ to $O(2^n)$. Note that the high number of puzzles, which is $N = 2^m$, become useless to prevent attack of Oscar, since Oscar can observe real index and realizes brute force attack directly to the selected puzzle. Sending index in clear text may give small advantages compared to the original MC such that Alice does not make a search for pseudo-index. However, this search effort is completely insignificant since Alice stores pseudo-indices sorted and finds a

corresponding real index easily. However, sending real index in clear text causes a significant security degradation that can not be compared with neglible search time advantage.

3 Improved Merkle Cryptosystem (IMC)

In this section, we present details of our Improved Merkle Cryptosystem (IMC). We make improvements over MC and VMC for three major points. Firstly, we increase security of the VMC by eliminating security problem which stems from sending real index in clear text. Notice that, computational advantages of Bob in VMC remain unchanged while security of the cryptosystem is increased. Secondly, we use auxiliary secret keys, which increase security of the hashed secret value transmitted over network for key agreement. This approach provides security advantages over VMC for transmitted packets over network. Thirdly, we show that IMC reduces storage requirements and bandwidth consumption of Bob and Alice.

Following additional notations are used:

H : Cryptographic hash function. This hash function should be a secure hash function such as SHA family [12] or a cryptographic hash function having variable length output property (this may provide advantage for different bandwidth requirements of communication). y_i : Auxiliary secret key which is used to increase bit length security of the hashed message transmitted over network, P_i^* : Public key which is generated using y_i auxiliary keys. K_{s_a} and K_{s_b} denote session keys, which are generated by Alice and Bob, respectively. h : Secret hashed vector where $h_i \in h$, for all i, $1 \leq i \leq N$ where $N = 2^m$. $PRNG$: Pseudo Random Number Generator.

IMC algorithm is described below:

1.Alice generates auxiliary secret keys y_i and puzzle pairs $P_i = E_{K_i}(X)$, $P_i^* = E_{K_i}(y_i)$ for $1 \leq i \leq N$ where $N = 2^m$. Alice sends (P_i, P_i^*, X) for all i to Bob and stores (K_i, y_i) pairs as secret key pairs.

2. Alice generates hashed secret key vector h, $h_i = H(K_i||y_i)$ for $1 \leq i \leq N$ where $N = 2^m$. Alice stores h as secret key vector. She can store vector h in two different ways. Details for storage of vector h are given in Section 4.2.

3. Bob obtains (P_i, P_i^*, X) for $1 \leq i \leq N$ where $N = 2^m$. Then he generates random keys l_j similar to VMC step 2 such that $while(v, search\ on\ P_i)\{l_j = PRNG(), v = E_{l_j}(X), move\ indices\}$. If $(P_i == E_{l_j}(X))$ then $K_i = l_j$ and Bob finds one of the secret keys K_i. Using K_i, Bob decrypts P_i^* and obtains secret auxiliary key $y_i = D_{K_i}(P_i^*)$.

4. Bob calculates $h' = H(K_i||y_i)$ and sends h' value to Alice. Notice that, only Alice knows K_i and y_i and using these secret key pairs, only Alice can calculate and verify h' value. Since one-way properties of H, Oscar can not find K_i and y_i from h'.

5. Session key agreement can be done with three different ways:

- *Alice decides session key:* Bob sends h' to Alice. Alice searches h' over vector h. If she finds then Alice and Bob agree on key $(K_i||y_i)$. Alice generates session key K_{s_s} and calculates $K_{s_s}{}' = E_{K_i||y_i}(K_{s_s})$ and sends $K_{s_s}{}'$ to Bob. Bob decrypts $K_{s_s}{}'$ and obtains $K_{s_s} = D_{K_i||y_i}(K_{s_s}{}')$.Alice and Bob agree on session key K_{s_s}.
- *Bob decides session key:* Bob generates K_{s_b} and calculates $K_{s_b}{}' = E_{K_i||y_i}(K_{s_b})$. Bob sends $(h', K_{s_b}{}')$ pair to Alice. Alice searches h' over vector h. If she finds then Alice and Bob agree on key $(K_i||y_i)$.Alice decrypts $K_{s_b}{}'$ and obtains $K_{s_b} = D_{K_i||y_i}(K_{s_b}{}')$.Alice and Bob agree on session key K_{s_b}.
- *Alice and Bob jointly decide session key:* Alice and Bob agree on $(K_i||y_i)$ similar to steps above and they exchange K_{s_s} and K_{s_b} session keys. They calculate their joint session key $K_s{}' = K_{s_s} \oplus K_{s_b}$.

4 Analysis and Comparison of IMC

In this section, we analyze properties of IMC and compare it to the MC and VMC. Also, we compare the security of IMC to the some well-known public key cryptosystems. Firstly, we analyze security properties and advantages of IMC over MC and VMC showing that IMC provides higher security than MC and VMC. Secondly, we present storage advantages of IMC over MC and mention some additional techniques to reduce storage requirements of IMC.

4.1 Security Analysis and Advantages of IMC

In IMC, in order to hide key agreement value, which is secret key $(K_i||y_i)$, we calculate hash of $(K_i||y_i)$, $h' = H((K_i||y_i))$ and transmit h' over network. Due to one-way property of cryptographic hash functions, Oscar can not find $(K_i||y_i)$ from h'. With our improvement, in order to obtain $(K_i||y_i)$, Oscar has to realize brute force attack to all puzzles ($N = 2^m$ puzzles). Since brute force attack to a single puzzle requires $O(2^n)$ computational effort, total computational effort of Oscar becomes $O(2^{n+m})$. In VMC, since real index is sent in clear text, Oscar knows which index Bob has chosen. Thus, computational effort of Oscar in VMC is only $O(2^n)$. In table 3, we can see security advantages of the IMC over VMC $(O(2^{n+m}) > O(2^n))$.

IMC can use larger key bit length for a single puzzle by shifting computational advantages of Bob to the overall system security (properties of VMC). Shifting computational advantages of Bob to the key bit length of a single puzzle (parameter for overall system security), we can select n such that $n > n'$. Thus, $O(2^{n+m}) > O(2^{n'+m})$ and IMC can provide higher security than MC using this approach. In table 3, advantages of IMC over MC can be seen. In addition to this, in table 3, security/performance advantage of IMC over MC and VMC is shown. It is calculated by dividing computational effort of Oscar to the computational effort of Bob. This gives us a criterion about the efficiency of the cryptosystem. We can see that both MC and VMC have $O(2^m)$ security/performance value while IMC has $O(2^{n+m})/O(2^{n-m}) = O(2^{2m})$ which is more efficient than MC and VMC.

Another improvement of IMC is that it uses auxiliary key y_i to increase bit length security of the hashed key agreement value h'. Suppose that Oscar obtain h' value by eavesdropping. In order to find $(K_i||y_i)$ from h', Oscar should try all possible $O(2^{2n})$ key space for detecting a one-to-one mapping among generated random keys and h' value. One-way properties of cryptographic hash function does not allow Oscar to recover $(K_i||y_i)$ from h' without brute force attack under the assumption of random behavior of hash functions [13]. Notice that, $|(K_i||y_i)| = 2n$ and for $n \simeq 70$ bits, $|(K_i||y_i)| = 140$ bits. This provides the security in the order of $O(2^{140})$. If only $h' = H(K_i)$ was used instead of $h' = H(K_i||y_i)$ then brute force effort of Oscar would have been $O(2^n)$. Under this condition, security of the transmitted message over network $(O(2^n))$ would have been lower than security of overall system cryptosystem ($O(2^{n+m})$) and Oscar would have broken system easily by attacking h' value instead of P_i puzzles. The main idea behind of the using auxiliary keys is preventing IMC from this attack. In VMC, messages transmitted over network are encrypted using only n bit K_i keys. Thus, IMC provides higher security for messages transmitted over network (including key agreement value) than that of the VMC. In MC, session keys are embedded into puzzle P_i. When Bob solves the puzzle, he uses session key to encrypt message, which is transmitted over network. Thus, message security of MC depends on key bit length of the session key and overall security of the cryptosystem. Results are summarized at Table 3.

4.2 Storage Analysis and Advantages of IMC

IMC has storage advantages over MC. In MC, a single puzzle P_i contains three components, which are S, r_i, and K_{s_i}, respectively (total t bits). These additional components increase bit length of a single puzzle and cause significant storage and transmission load. However, in IMC, there are puzzle pairs (P_i, P_i^*) each of them having $2n$ bit length. Thus, for $N = 2^m$ puzzles, IMC provides $O(2^m t - 2^{m+1}n) = O(2^m(t - 2n))$ storage advantages over MC. For example, bit length of a single puzzle in MC with 40 bit redundancy, 40 bit pseudo-index and 128 bit session key are approximately $t \approx 208$ bit. In IMC, the bit length of a key can be selected up to 70 bits (due to storage and computational limits). Thus, bit length of a single puzzle pair is $2n \approx 140$ *bits*. Consequently, for $m \approx 30$, IMC provides storage advantages up to $(2^{30} * 68) \simeq 1$ GB for these settings when compared to MC. Important point is that, same amount of gain is also obtained for network bandwidth consumption. Notice that, VMC has a small storage advantages when compared to IMC (IMC : $O(2 * 2^m n)$, VMC: $O(2^m n)$). However, for corresponding small storage load, IMC has significant security advantages over VMC. These results can also be observed in table 3.

Apart from these, in IMC step 2, we have discussed that secret key vector h can be stored in two different ways. This is a tradeoff approach among storage and computational resources of Alice. If Alice has sufficient storage resources, she stores vector h permanently. Then, whenever a key agreement occurs, Alice directly searches h' over vector h for key agreement. This approach provides

computational resource advantage. However, if Alice does not have sufficient storage capability, for each key agreement, she dynamically generates h_i elements using (K_i, y_i) secret key pairs and compares h_i with h' to find a match. Thus, Alice does not have to store vector h permanently. Since cryptographic hash functions are fast, with feasible amount of puzzle ($N = 2^m, m \approx 30$), search operation becomes feasible. This approach provides storage advantage.

Table 3. Comparion of IMC to MV and VMC

		MC	VMC	IMC
Computational Complexity	Alice	2^m	2^m	2^m
	Bob	2^n	2^{n-m}	2^{n-m}
	Oscar	2^{n+m}	2^n	2^{n+m}
Storage Complexity	Alice	$2^m t$	$2^m n$	$2^{m+1} n$
	Bob	$2^m t \rightarrow 1$	$2^m n \rightarrow 1$	$2^{m+1} n \rightarrow 1$
	Oscar	$2^m t$	$2^m n$	$2^{m+1} n$
Security Comparison		2^{n+m}	2^n	2^{n+m}
Security/Computational		2^m	2^m	2^{2m}
Message Security		$\|K_s\|$	2^n	2^{2n}

4.3 Comparison of IMC with MC, VMC and Some Well-Known Public Key Cryptosystems

Table 4 demonstrates comparison of the IMC with MC, VMC and some well-known public key cryptosystems. Symmetric Cryptosystem Bit Length (SCBL) security gives total bit length strength of the MC, VMC and IMC to resist attack of Oscar. For example, 100 bits mean that computational effort of Oscar to break cryptosystem is equivalent to break 100 bits block cipher. Note that, it does not mean that bit length of the key that will be used for block cipher is 100 bits, but total effort (using all puzzles in the system) corresponds to 100 bits security. To reach this security level, parameters $m \simeq 30$ bits, $n' = 40$ bits and $n = 70$ bits are selected for today's and near future feasible memory and computational possibilities. Brute force attack capability of Bob is selected as 2^{40} that allows feasible search time for key agreement. Storage capability of Alice and Bob is selected as approximately $2^{30} \cdot 140$ bits so that it is feasible for current hardware possibilities. Using these parameters, maximum security available for the MC is 2^{70}. In IMC, using aforementioned improvements, security level can be reached up to 2^{100} bits ($O(2^{n+m})$) that extends approximate lifespan of the cryptosystem to 30 years (Table 4) [14]. For these parameters, providing more than 70 bit security becomes infeasible both for MC and VMC. Remainder parts of the table 4 shows equivalent bit length security level for various public key cryptosystems and their related lifespan and economical cost values. Corresponding values for symmetric key bit length security are obtained from [14]. For these comparisons, [15] can also be used. With these interpretations, we see that IMC can provide as high security as some well-known public key

cryptosystems. In table 4, following abbreviations are used: $PKCL$: Public Key Cryptography bit Length. CAS: Classical Asymmetric Cryptography like RSA. $SDLF$: Sub Group Discrete Logarithm problem Field. EC: Elliptic Curve. LB: Lower Bound.

Table 4. Comparison of IMC with VMC-MC and some well-known public key cryptosystems for various criteria

SCBL	MC	70	Infeasible for Participants				
	VMC	70	Infeasible for Participants				
	IMC	70	76	82	88	94	100
PKCL	CAS	952	1279	1613	2054	2560	3137
	SDLF	704	960	1248	1632	2080	2592
	SDL Key Size	125	135	145	156	167	178
	EC Size	132	155	173	197	218	240
Infeasible Number of MIPS Years		$8{\cdot}10^{9}$	$5{\cdot}10^{11}$	$2{\cdot}10^{13}$	$2{\cdot}10^{15}$	$1{\cdot}10^{17}$	$8{\cdot}10^{18}$
LB for HW attack cost for 1 day breaking		$1{\cdot}10^{8}$	$3{\cdot}10^{8}$	$4{\cdot}10^{8}$	$7{\cdot}10^{8}$	$1{\cdot}10^{9}$	$2{\cdot}10^{9}$
Corresponding Lifespan		2000	2008	2015	2023	2031	2039

5 Conclusion and Future Works

In this study, we propose Improved Merkle Cryptosystem (IMC), which can be considered as an alternative method for key agreement schemes, based on only symmetric cryptosystem and cryptographic hash functions without requiring a Trusted Third Part (TTP). As a novelty, IMC uses cryptographic hash functions and auxiliary keys to increase security of MC and VMC. Unlike VMC, IMC hides key agreement value using cryptographic hash functions and enhances the security of key agreement value utilizing auxiliary keys. These approaches provide significant security advantages over VMC. Since IMC utilizes some advantages of VMC over MC, IMC also provides higher security than MC. Different puzzle structure of IMC reduces storage requirement of the cryptosystem when compared to MC. Our improvements provide a solution to use MC for long term security, which is compatible with some well-known public key cryptosystems, within today's feasible hardware possibilities.

MC does not provide security against active attacks such as message replay and injection attacks. As a future work, we consider using IMC to develop a key agreement scheme, which can provide major cryptographic goals such as confidentiality, integrity, authentication and unforgeability together. In order to this, we consider using some principles of signcryption [16]. We will integrate IMC with a signcryption based key exchange schemes [17], which uses nonce and time-stamps to prevent cryptosystem from active attacks. We believe that, this integrated cryptosystem, Signcryption Type Authentic Key Establishment scheme (STAKE), will solve active attack problems of IMC and will provide additional cryptographic goals.

Acknowledgements

This work is supported by the State Planning Organization of Turkey under "Next Generation Satellite Networks Project", and Bogaziçi University Research Affairs.

References

1. W. Diffie and M. E. Hellman. New Directions in Cryptography, IEEE Trans. Information Theory, vol. IT-22, Nov. 1976, pp: 644 654.
2. Ueli Maurer. Cryptography 2000 -10 Years Back, 10 Years Ahead, Lecture Notes in Computer Science, Springer-Verlag, vol. 2000, pp.63-85, 2001.
3. Standard specifications for public key cryptography. IEEE P1363/D13, November 1999.
4. D. Johnson, A. Menezes. The Elliptic curve digital signature algorithm (ECDSA)", February 24, 2000.
5. J. Hoffstein, J. Pipher, and J.H. Silverman, NTRU: A Ring-Based Public Key Cryptosystem, Proceedings of ANTS III, Portland, June 1998.
6. R. C. Merkle. Secure Communications over Insecure Channels, Communications of the ACM 21(4), pp294–299 (April 1978).
7. Chris Mitchell. Public key encryption using block ciphers, technical report RHUL-MA-2003-6, 9 September (Department of Mathematics, Royal Holloway, University of London), 2001.
8. R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto, MIT LCS Tech. Report MIT/LCS/TR-684, 1996.
9. Aldar C.-F. Chan, Ian F. Blake. Scalable, Server-Passive, User-Anonymous Timed Release Cryptography, icdcs, pp. 504-513, 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), 2005.
10. D. Dean and A. Stubblefield. Using client puzzles to protect TLS, Proceedings of the USENIX Security Symposium, August 2001.
11. NIST. Specifications for the Advanced Encryption Standard(AES). Federal Information Processing Standards Publications (FIPS PUB) 197, November 2001. U.S. Department of Commerce, N.I.S.T.
12. NIST. Secure Hash Standard. Federal Information Processing Standards Publications(FIPS PUB) 180-2, August 26, 2002. U.S. Department of Commerce, N.I.S.T.
13. D. Stinson. Cryptography Theory and Practice. CRC Press, Inc., Third Edition, 2005.
14. Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes, Journal of Cryptology, 14(4):255–293, 2001.
15. A.K. Lenstra. Unbelievable security, Proceedings Asiacrypt 2001, LNCS 2248, Springer-Verlag 2001, 67-86.
16. Y. Zheng. Digital signcryption or how to achieve Cost(Signature Encryption) << Cost(Signature) + Cost(Encryption). Advances in Cryptology – Crypto'97, Lecture Notes in Computer Science, Vol. 1294, pp. 165-179, Springer-Verlag, 1997.
17. Y. Zheng. Shortened digital signature, signcryption, and compact and unforgeable key agreement schemes (A contribution to IEEE P1363 Standard for Public Key Cryptography), July 1998.